# Reconocimiento de Patrones

Version 2023-2

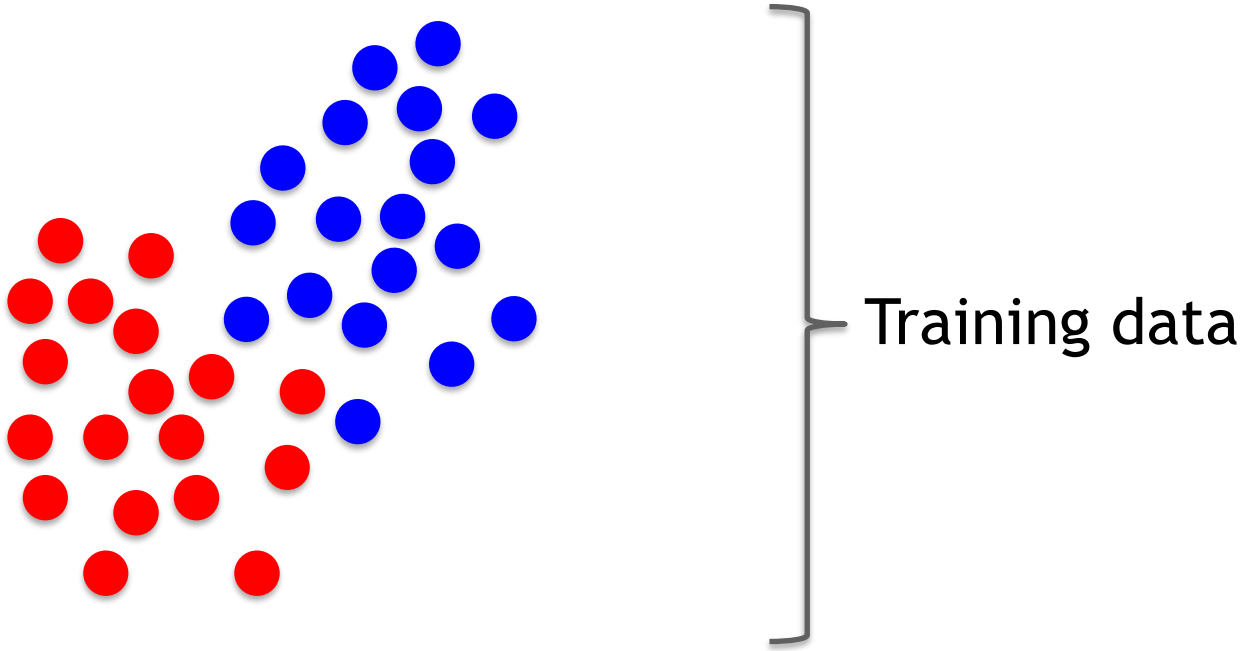
## KNN

[ Capítulo 4 ]

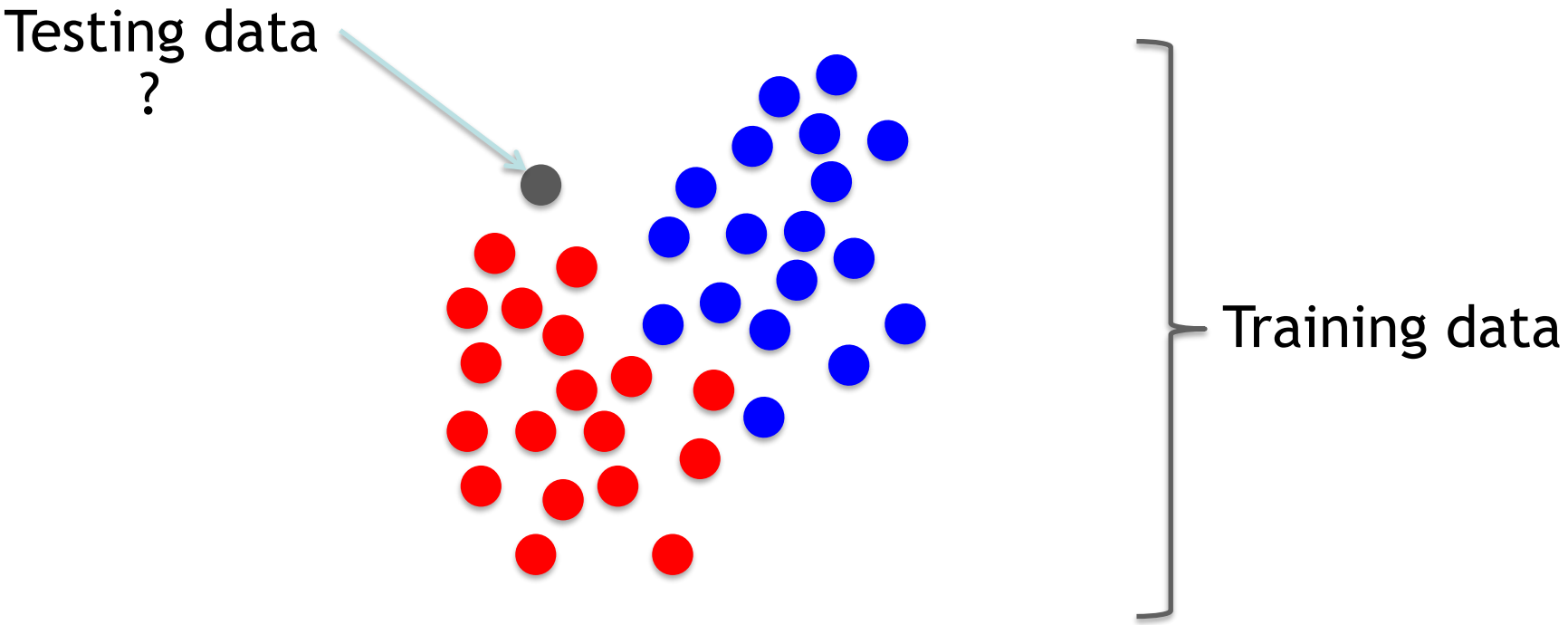
Dr. José Ramón Iglesias
DSP-ASIC BUILDER GROUP
Director Semillero TRIAC
Ingenieria Electronica
Universidad Popular del Cesar

# KNN: k nearest neighbors

# KNN: k nearest neighbors (two classes)



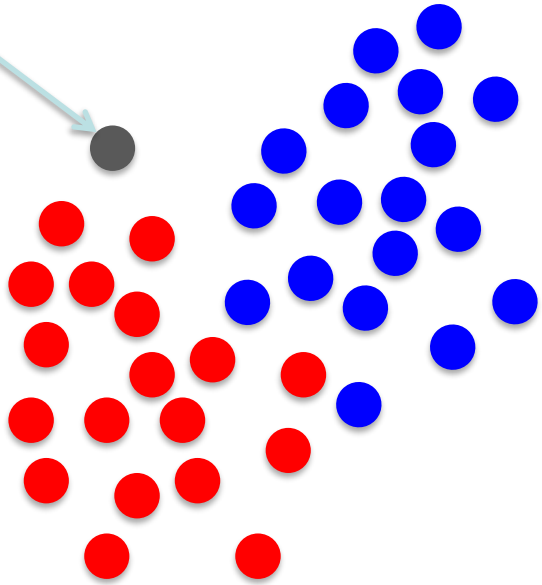
# KNN: k nearest neighbors (two classes)



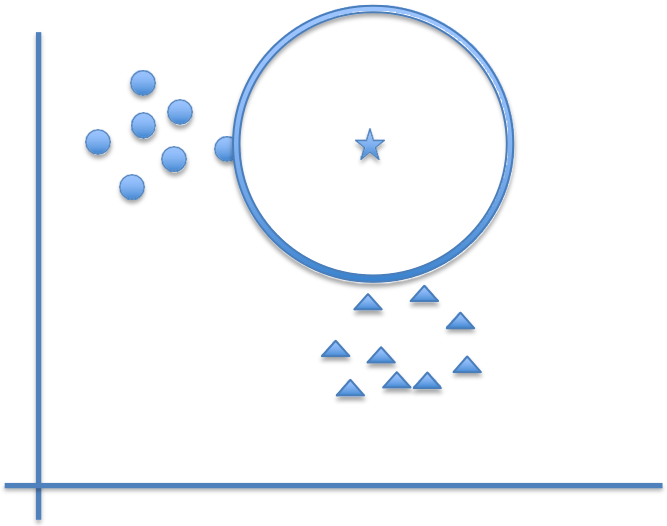
# KNN: k nearest neighbors (two classes)

**KNN Algorithm**

Testing data



# K- Vecinos Más Cercanos, Clasificación



Para clasificar una nueva muestra z a partir del conjunto de entrenamiento $x_i$ .

1. Se hallan las distancias* de $z$ a todos los $x_i$
2. Se toman las menores k distancias.
3. Se selecciona la clase a partir del $y_i$ asociado a los $x_i$ de menor distancia con z .

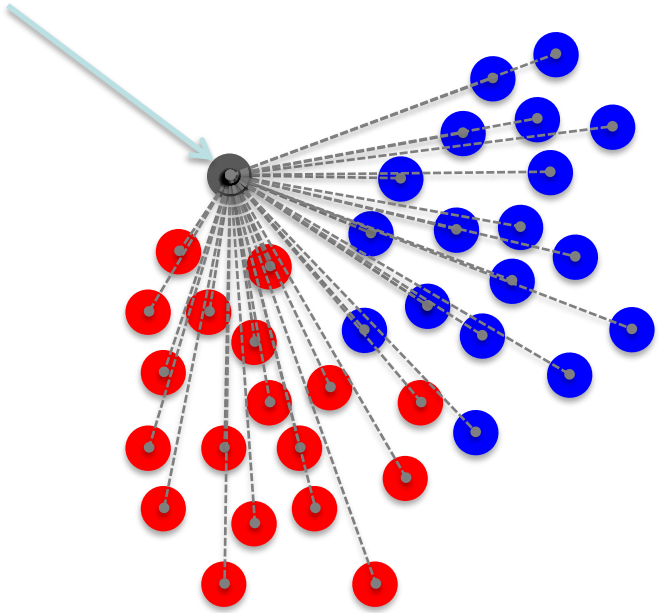
Si k=1 se crea una partición de Voronoi.

La selección puede hacerse a partir de la moda “el de mayor votación”

*Por distancia se hace referencia a cualquier métrica, por ahora usemos la distancia euclidiana

# KNN: k nearest neighbors (two classes)

Testing data



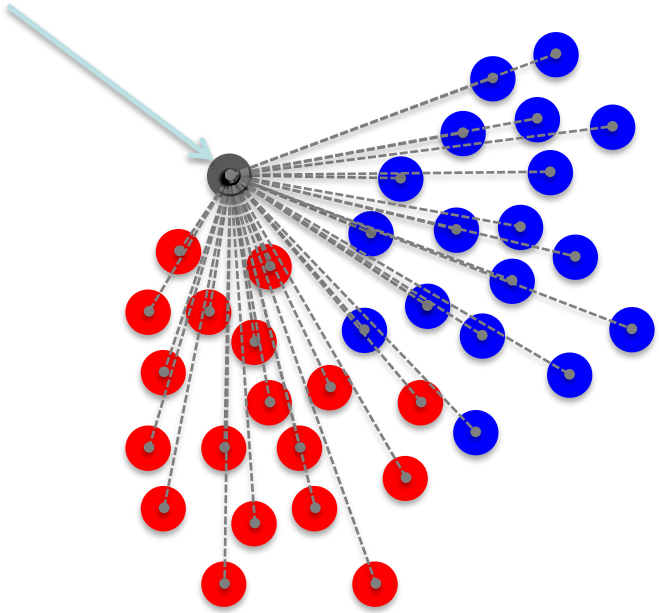
**KNN Algorithm**

1. Distances

0.3655
0.7015
0.6712
0.7474
0.4313
0.2880
0.0115
0.9202
0.5304
0.9362
0.5989
0.9447
0.0569
0.3264
0.6811
0.1332
0.0226
0.2435
0.0705
0.4839
0.3631
0.1090
0.6296
0.0508
0.7660
0.9544
0.6487
0.1519
0.7936
0.9525

# KNN: k nearest neighbors (two classes)

Testing data

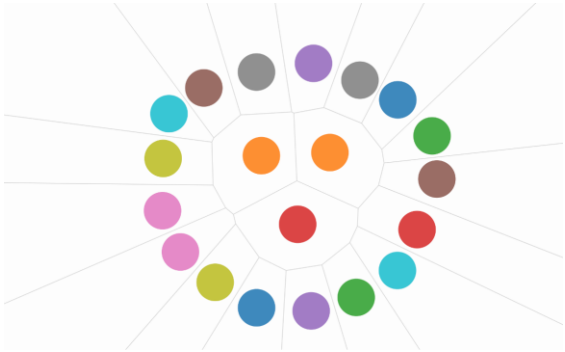


**KNN Algorithm**

1. Distances
2. Sort

| | |
|---|---|
| 0.3655 | 0.0115 |
| 0.7015 | 0.0226 |
| 0.6712 | 0.0508 |
| 0.7474 | 0.0569 |
| 0.4313 | 0.0705 |
| 0.2880 | 0.1090 |
| 0.0115 | 0.1332 |
| 0.9202 | 0.1519 |
| 0.5304 | 0.2435 |
| 0.9362 | 0.2880 |
| 0.5989 | 0.3264 |
| 0.9447 | 0.3631 |
| 0.0569 | 0.3655 |
| 0.3264 | 0.4313 |
| 0.6811 | 0.4839 |
| 0.1332 | 0.5304 |
| 0.0226 | 0.5989 |
| 0.2435 | 0.6296 |
| 0.0705 | 0.6487 |
| 0.4839 | 0.6712 |
| 0.3631 | 0.6811 |
| 0.1090 | 0.7015 |
| 0.6296 | 0.7474 |
| 0.0508 | 0.7660 |
| 0.7660 | 0.7936 |
| 0.9544 | 0.9202 |
| 0.6487 | 0.9362 |
| 0.1519 | 0.9447 |
| 0.7936 | 0.9525 |
| 0.9525 | 0.9544 |

# Consideraciones ☺



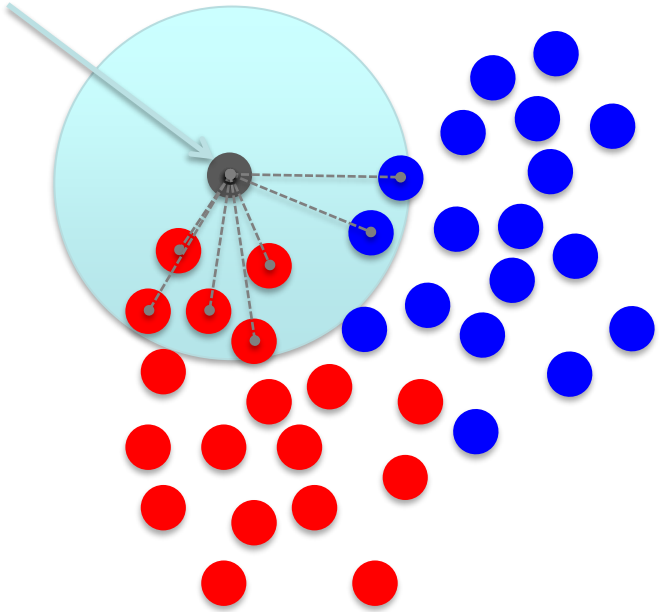
Para 2 clases se debe tomar un k impar.

- Solo es necesaria 1 muestra por clase para lograr un clasificador por KNN.
- Se debe seleccionar cuidadosamente el k.
- K no debe ser un múltiplo del número de clases.
- e.g. si hay 6 clases se puede tomar K=6.
- KNN es considerada una técnica No paramétrica en sentido estadístico.

- KNN es clasificado como un “lazy learner” ya que no estima una función de frontera de decisión.

# KNN: k nearest neighbors (two classes)

Testing data



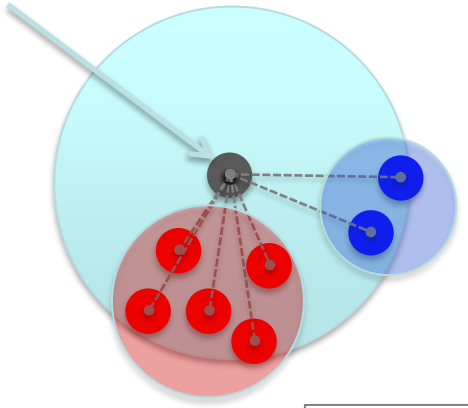
**KNN Algorithm**

1. Distances
2. Sort
3. Take the k nearest
   (example k=7)

0.0115
0.0226
0.0508
0.0569
0.0705
0.1090
0.1332
0.1519
0.2435
0.2880
0.3264
0.3631
0.3655
0.4313
0.4839
0.5304
0.5989
0.6296
0.6487
0.6712
0.6811
0.7015
0.7474
0.7660
0.7936
0.9202
0.9362
0.9447
0.9525
0.9544

# KNN: k nearest neighbors (two classes)

Testing data



2 blue

5 red → Testing sample is red

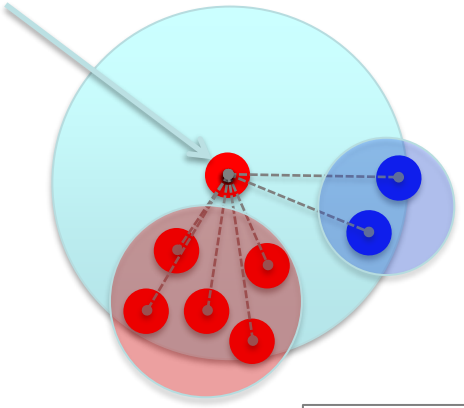
**KNN Algorithm**

1. Distances
2. Sort
3. Take the k nearest (example k=7)
4. Majority vote

0.0115
0.0226
0.0508
0.0569
0.0705
0.1090
0.1332
0.1519
0.2435
0.2880
0.3264
0.3631
0.3655
0.4313
0.4839
0.5304
0.5989
0.6296
0.6487
0.6712
0.6811
0.7015
0.7474
0.7660
0.7936
0.9202
0.9362
0.9447
0.9525
0.9544

# KNN: k nearest neighbors (two classes)

Testing data



2 blue

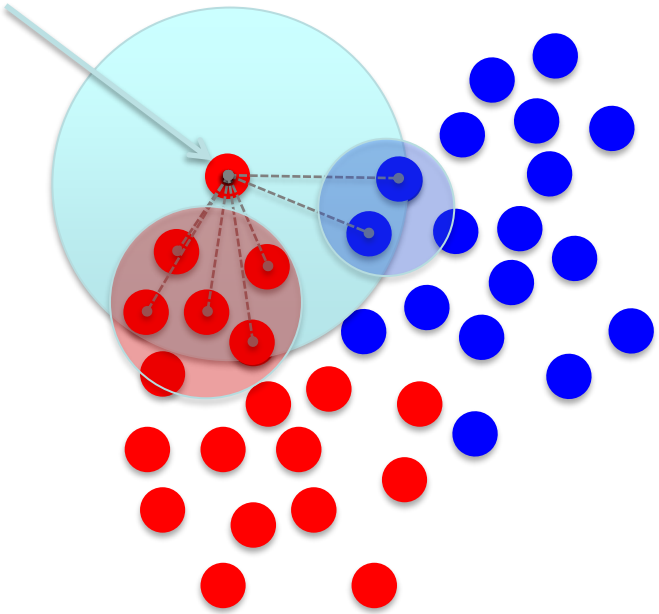
5 red → Testing sample is red

**KNN Algorithm**

1. Distances
2. Sort
3. Take the k nearest
   (example k=7)
4. Majority vote

# KNN: k nearest neighbors (two classes)

Testing data



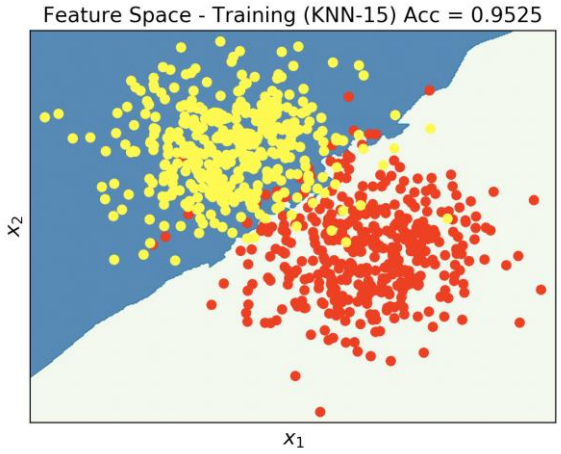
**KNN Algorithm**

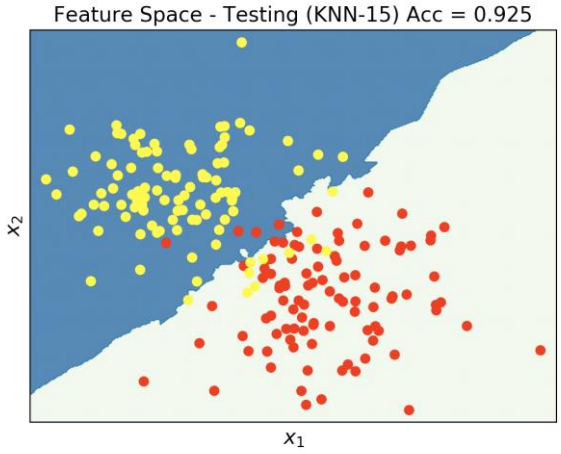
1. Distances
2. Sort
3. Take the k nearest
   (example k=7)
4. Majority vote

TRAINING

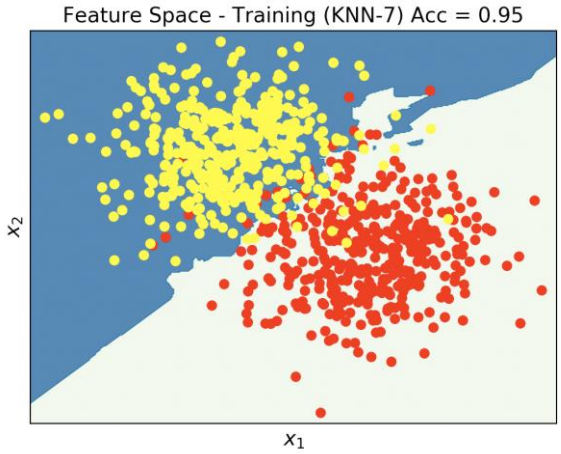
TESTING

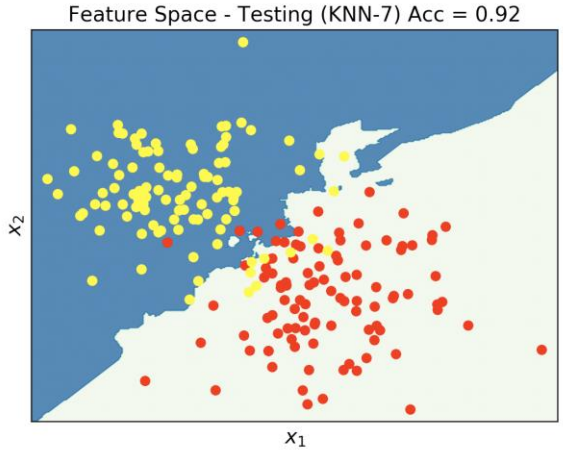
k = 15




k = 7

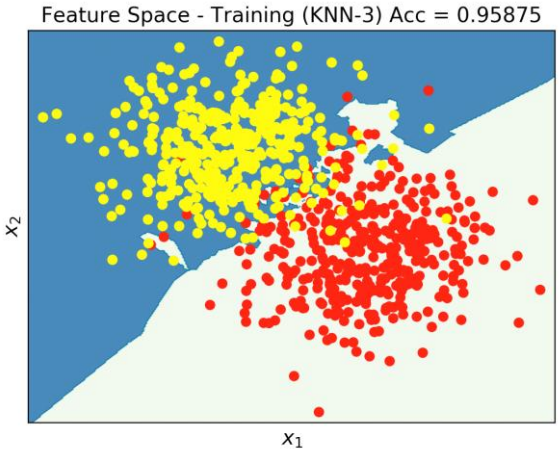


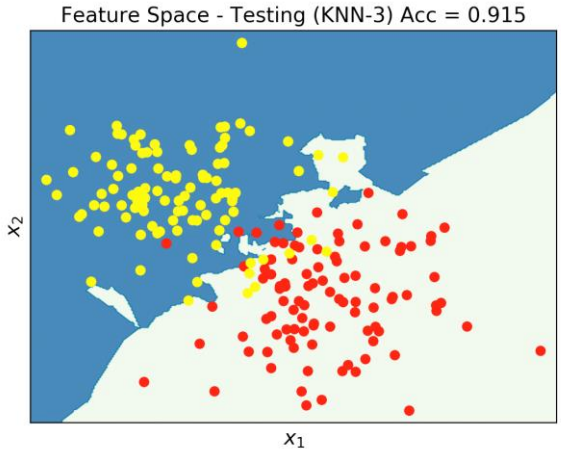



TRAINING

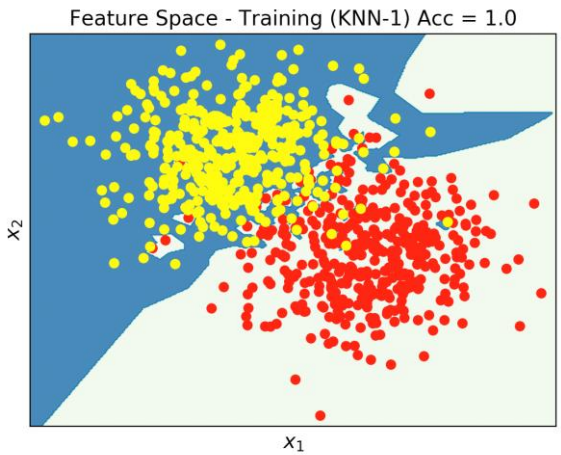
TESTING

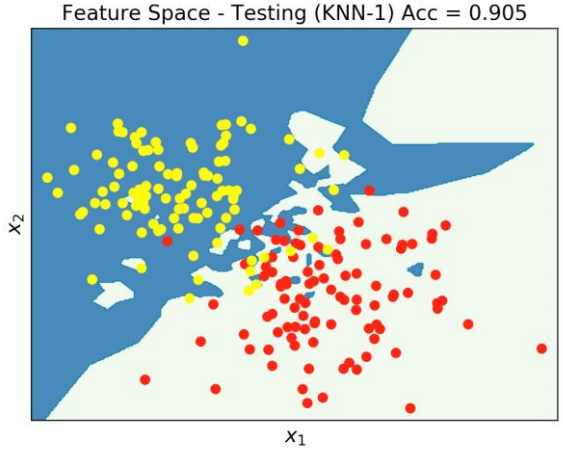
k = 3




k = 1



# Problemas ☹

- Si se toman muchas muestras $x_i$ la clasificación puede llegar a ser lenta.
- No existe un método estándar para determinar un valor óptimo para k.
  - Valores pequeños de k son susceptibles a afectaciones por valores fuera de tendencia “Ruido”
  - Valores Grandes de k son más inmunes a ruido pero si k es muy grande las categorías con pocas muestras pueden llegar a no ser seleccionadas nunca.
  - Si se selecciona mal el k se puede llegar fácilmente a “grandes” regiones incongruentes empatadas en votación.

# Thumb rules!
# Selección del K


Tomado de Quora

Puede usarse k como el número impar más cercano a la raíz cuadrada de n donde n es:

- El número total de datos $x_i$
- El número de datos en la clase más pequeña.

# Thumb rules! Cuando usar KNN

- Cuando los datos están bien etiquetados.
  - No hay ruido* aparente en las etiquetas.
- Cuando hay pocos datos.
  - Lo contradictorio es que KNN funciona mejor si hay muchos datos!
- Cuando no les están pagando bien por el trabajo y quieren terminar rápido y entregar algo que funcione.

*Por ejemplo bases de datos hechas a partir de encuestas pueden tener ruido

# Regresión por KNN

KNN puede ser usado para hacer regresión.

Se puede usar el promedio* de $y_i$ para los K-vecinos más cercanos al punto $z$.

*Puede usarse en realidad cualquier estimador del valor esperado.

# Mejoras a KNN

Puede implementarse una mejora tanto para regresión como para clasificación por KNN.

El valor estimado de clasificación o regresión a partir de $y_i$ se hace pesando el estimador a partir de las distancias a $z$

# Ejercicio en clase

Realizar un ejemplo usando python del método de KNN para clasificación.

Experimentar con diferentes configuraciones.